IN THE CLAIMS:

Please add claims 16-20.

Please amend the claims as shown in the following claims listing.

1. (Currently amended) A system for controlling co-scheduling of processes in a computer comprising at least one process and a spin daemon, the at least one process being configured to, when it is waiting for a flag to change condition, transmit a flag monitor request to the spin daemon and to de-schedule itself by removing an identifier associated with the at least one process from a task scheduling list accessed by an operating system and to notify the operating system of the removal, the spin daemon being configured to, after receiving the flag monitor request monitor the flag and, ~~after~~ in response to determining the flag ~~changes~~ has changed condition, enable the at least one process to be re-scheduled by the operating system for execution by the computer.

2. (Original) A system as defined in claim 1 in which said spin daemon is configured to monitor a plurality of flags, each in response to a flag monitor request, the spin daemon maintaining a list identifying those flags it is to monitor, the spin daemon being further configured to, when it receives a flag monitor request, add an identification of a flag associated with the request to the list.

3. (Original) A system as defined in claim 2 in which said flags are contained in a memory segment, the spin daemon being configured to enable the at least one process to be re-scheduled following a change of condition of any flag in said memory segment.

4. (Original) A system as defined in claim 3 in which said at least one process is configured to register with said spin daemon during registration the at least one process being configured to provide the spin daemon with an identifier for the memory segment, the spin daemon being configured to provide a handle, the at least one process being configured to use the handle in the flag monitor request.

5. (Original) A system as defined in claim 1 in which said at least one process and said spin daemon are configured to communicate over a socket.

6. (Currently amended) A method of controlling co-scheduling of processes in a computer comprising at least one process and a spin daemon, the method comprising the steps of:

A. enabling the at least one process to, when it is waiting for a flag to change condition, transmit a flag monitor request to the spin daemon and to de-schedule itself by removing an identifier associated with the at least one process from a task scheduling list accessed by an operating system and notifying the operating system of the removal,

B. enabling the spin daemon to, after receiving the flag monitor request monitor the flag and, ~~after~~ in response to determining the flag ~~changes~~ has changed condition, enable the at least one process to be re-scheduled by the operating system for execution by the computer.

7. (Original) A method as defined in claim 6, the spin daemon being configured to monitor a plurality of flags, each in response to a flag monitor request, the spin daemon maintaining a list identifying those flags it is to monitor, the method including the step of enabling the spin daemon being to, when it receives a flag monitor request, add an identification of a flag associated with the request to the list.

8. (Original) A method as defined in claim 7 in which said flags are contained in a memory segment, the method including the step of enabling the spin daemon to enable the at least one process to be re-scheduled following a change of condition of any flag in said memory segment.

9. (Currently amended) A method as defined in claim 8 further including the steps of

A enabling the at least one process to register with said spin daemon, during registration the at least one process being configured to provide the spin daemon with an identifier for the memory segment; and

B. enabling the spin daemon to provide a handle for use by the at least one process in the flag monitor request.

10. (Original) A method as defined in claim 6 further comprising the step of enabling the at least one process and said spin daemon to communicate over a socket.

11. (Currently amended) A computer program product for use in connection with a computer to control co-scheduling of at least one process in the computer, the computer program product including a computer readable medium having encoded thereon program instructions comprising:

A. a process module configured to enable the computer to, when the at least one process is waiting for a flag to change condition, transmit a flag monitor request and de-schedule itself by removing an identifier associated with the at least one process from a task scheduling list accessed by an operating system and notifying the operating system of the removal,

B. a spin daemon module configured to enable the computer to, after receiving the flag monitor request, monitor the flag and, ~~after~~ in response to determining the flag ~~changes~~ has changed condition, enable the at least one process to be re-scheduled by an operating system for execution by the computer.

12. (Original) A computer program product as defined in claim 11 in which said spin daemon is configured to enable the computer to monitor a plurality of flags, each in response to a flag monitor request, the spin daemon enabling the computer to maintain a list identifying those flags it is to monitor, the spin daemon being further configured to enable the computer to, when it receives a flag monitor request, add an identification of a flag associated with the request to the list.

13. (Currently amended) A computer program product as defined in claim 12 in which said flags are contained in a memory segment, the spin daemon being configured to enable the computer to enable the at least one process to be re-scheduled following a change of condition of any flag in said memory segment.

14. (Original) A computer program product as defined in claim 13 in which said at least one process is configured to enable the computer to register with said spin daemon, during registration the at least one process being configured to enable the computer to provide the spin daemon with an identifier for the memory segment, the spin daemon being configured to enable the computer to provide a handle, the at least one process being configured to use the handle in the flag monitor request.

15. (Original) A computer program product as defined in claim 11 in which said at least one process and said spin daemon are configured to enable the computer to communicate over a socket.

16. (New) A system as defined in claim 1, wherein the spin daemon is a low-priority process having a lower processing priority than the at least one process.

17. (New) A system as defined in claim 1, wherein, in response to receiving a notification from the spin daemon, the at least one process is configured to enable itself to be re-scheduled for execution by requesting the operating system to load the identifier into the task scheduling list.

18. (New) A method as defined in claim 6, wherein the spin daemon is a low-priority process having a lower processing priority than the at least one process.

19. (New) A method as defined in claim 6, further comprising, in response to receiving a notification from the spin daemon, enabling the at least one process to enable itself to be re-scheduled for execution by requesting the operating system to load the identifier into the task scheduling list.

20. (New) A computer program product as defined in claim 11, wherein process module is further configured to, in response to receiving a notification from the spin daemon, enable itself to be re-scheduled for execution by requesting the operating system to load the identifier into the task scheduling list.